import cv2

import numpy as np

from tensorflow.keras.applications import MobileNetV2

from tensorflow.keras.applications.mobilenet\_v2 import preprocess\_input

from tensorflow.keras.layers import Dense, Dropout, Flatten, Input

from tensorflow.keras.models import Model

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Function to train the mask detection model

def train\_mask\_detector(dataset\_path, output\_model\_path):

# Load pre-trained MobileNetV2 model without top layers

base\_model = MobileNetV2(weights='imagenet', include\_top=False, input\_tensor=Input(shape=(224, 224, 3)))

# Add custom layers on top of MobileNetV2

head\_model = base\_model.output

head\_model = Flatten(name="flatten")(head\_model)

head\_model = Dense(128, activation="relu")(head\_model)

head\_model = Dropout(0.5)(head\_model)

head\_model = Dense(2, activation="softmax")(head\_model)

# Combine base model with custom head

model = Model(inputs=base\_model.input, outputs=head\_model)

# Freeze base layers

for layer in base\_model.layers:

layer.trainable = False

# Compile the model

model.compile(optimizer=Adam(lr=1e-4), loss='binary\_crossentropy', metrics=['accuracy'])

# Data preprocessing

train\_datagen = ImageDataGenerator(

preprocessing\_function=preprocess\_input,

rotation\_range=20,

zoom\_range=0.15,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

shear\_range=0.15,

horizontal\_flip=True,

fill\_mode="nearest")

train\_generator = train\_datagen.flow\_from\_directory(

dataset\_path,

target\_size=(224, 224),

batch\_size=32,

class\_mode='categorical')

# Train the model

model.fit(

train\_generator,

steps\_per\_epoch=train\_generator.samples // train\_generator.batch\_size,

epochs=20)

# Save the trained model

model.save(output\_model\_path)

# Function to predict mask on an image

def predict\_mask(model\_path, image\_path):

from tensorflow.keras.models import load\_model

# Load the trained model

model = load\_model(model\_path)

img = cv2.imread(image\_path)

img = cv2.resize(img, (224, 224))

img = np.expand\_dims(img, axis=0)

img = preprocess\_input(img)

pred = model.predict(img)

return "With Mask" if np.argmax(pred) == 0 else "Without Mask"

# Example usage for training

dataset\_path = 'dataset/train'

output\_model\_path = 'mask\_detector\_model.h5'

train\_mask\_detector(dataset\_path, output\_model\_path)

# Example usage for prediction

image\_path = 'test\_image.jpg'

result = predict\_mask(output\_model\_path, image\_path)

print(result)